Blockstack: Design and Implementation of a Global Naming System with Blockchains

Muneeb Ali[⋆†], Jude Nelson[⋆†], Ryan Shea[†], Michael J. Freedman[⋆]
[⋆]*Princeton University,* [†]*Onename*

Abstract

Cryptocurrency blockchains like Bitcoin and Namecoin and their respective P2P networks have seen significant adoption in the past few years, and show promise as naming systems with no trusted parties. Users can register human-readable names and securely associate data with them; only the owner of a particular private-key can write or update the name/value pair. In theory, many decentralized systems can be built using these cryptocurrency networks, such as new, decentralized versions of DNS or PKI. As the technology is relatively new and evolving rapidly, however, little production data or experience is available to guide design tradeoffs.

In this paper, we describe our experience operating a large, real-world deployment of a decentralized PKI service, called *Blockchain ID*, built on top of the Namecoin blockchain. We present various challenges (network reliability, throughput, security issues) that we needed to overcome while registering/updating over 33,000 entries on the blockchain which involved over 200,000 transactions on the Namecoin network. Further, we discuss how our experience informed the design of a new blockchain-based naming system, Blockstack. We detail why we changed from Namecoin to the Bitcoin network for the new system, as well as operational lessons from this migration. Blockstack is released as open source and currently powers a production PKI system for 40,000 users.

1 Introduction

Cryptocurrency blockchains and their respective P2P networks are useful beyond exchanging money. They provide cryptographically auditable, append-only ledgers that are already being used to build new, *decentralized* versions of DNS [37] and public-key infrastructure (PKI) [39], along with other applications like file storage [21] and document timestamping [18]. Because blockchains have no central points of trust or failure, they enable a new class of decentralized applications and services that minimize the degree to which users need to put complete trust in a single party, like a DNS root server or a root certificate authority.

Blockchain networks have attracted a lot of interest from enthusiasts, engineers, and investors; 955 million USD has been invested in blockchain startups in the last 3 years [17]. With the rapid capital infusion, infrastructure for blockchains is getting quickly deployed [16] and blockchains are emerging as publicly-available common infrastructure for building decentralized systems and applications. However, blockchain networks are at a very early stage and there is very little production data available to guide design trade-offs.

Many non-financial applications of blockchains imply the need for a naming system that securely binds *names*, which can be human-readable, to arbitrary *values*. The blockchain gives consensus on the global state of the naming system and provides an append-only global log for state changes. Writes to name/value pairs can only be announced in new blocks, as appends to the global log. The global log is logically centralized (all nodes on the network see the same state), but organizationally decentralized (no central party controls the log).

The decentralized nature of blockchain-based naming introduces meaningful security benefits, but certain aspects of contemporary blockchains present technical limitations. Individual blockchain records are typically on the order of kilobytes [45] and cannot hold much data. Latency of creating/updating records is capped by the blockchain's write propagation and leader election protocol, and it is typically on the order of 10-40 minutes [13]. The total new operations in each round are limited by average bandwidth of nodes participating in leader election (for Bitcoin the current bandwidth per new round is 1MB). Further, new nodes need to independently audit the global log from the beginning, which implies that as the system makes forward progress, the time to bootstrap new nodes increases linearly.

We believe that in spite of these scalability and performance challenges, blockchains provide important infrastructure for building secure, decentralized services. The cost of tampering with blockchains grows with their adoption; today, it would require hundreds of millions of dollars to attack a large blockchain like Bitcoin [1].

These benefits motivated us to use blockchains to build a new decentralized PKI system, called *Blockchain ID*. Our system enables users to register unique, human-readable usernames and associate public-keys, like PGP [49], along with additional data to these usernames. There is no need for any central or trusted party in our PKI system. This paper presents our experiences from operating this PKI system on the Namecoin network, which is one of the largest services built on top of a blockchain to date. We outline the challenges that we had to overcome for registering/updating over 33,000 user entries and for sending over 200,000 transactions on the Namecoin network.

Our production deployment led to many interesting experiences where we observed and analyzed network anomalies and security problems that were not discovered or documented before. **We discovered a critical security problem where a single miner consistently had more than 51% relative compute power, called *mining power*, on Namecoin** (see [32] for details on the 51% attack and mining power). A 51% attack is one of the most serious attacks on a blockchain and impacts the security and decentralization properties.

Moreover, we also encountered chronic networking issues with broadcasting transactions on the Namecoin network. Reliability of the network generally depends on how actively a blockchain network is monitored and maintained, as well as financial incentive for operating the network. Therefore, for both security and reliability reasons, blockchain-based services should use the largest and most-secure blockchain, which is currently Bitcoin.

Our experience with Namecoin informed the design and implementation of a new blockchain-based naming system, called Blockstack, that uses the Bitcoin blockchain. Unlike previous blockchain-based systems, Blockstack *separates its control- and data-plane considerations*: it keeps only minimal metadata (namely, data hashes and state transitions) in the blockchain and uses external datastores for actual bulk storage. Blockstack enables fast bootstrapping of new nodes by using checkpointing and *skip lists* to limit the set of blocks that a new node must audit to get started. We have released Blockstack as open source [12].

2 Motivation and Background

In this section, we describe the motivation for building naming systems that have no central point of trust and provide the relevant background on blockchains. In this paper, we use the term *naming system* to mean a) names are **human-readable** and can be picked by humans, b) name/value pairs have **strong sense of ownership**—that is, they can be owned by cryptographic keypairs, and c) there is **no central trusted party** or point of failure. Building a naming system with these three properties was considered impossible according to Zooko's Triangle [30] and most naming systems provide two out of these three properties [29]. Namecoin [37] used a blockchain-based approach to provide the first naming system that offered all three properties: human-readability, strong ownership, and decentralization.

2.1 Background on Blockchains

Blockchains provide a global append-only log that is publicly writeable. Writes to the global log, called *transactions*, are organized as *blocks* and each block packages multiple transactions into a single atomic write. Writing to the global log requires a payment in the form of a *transaction fee*. Nodes participating in a blockchain network follow a leader-election protocol for deciding which node gets to write the next block and collect the respective transaction fees. Not all nodes in the network participate in leader election. Nodes actively competing to become the leader of the next round are called *miners*. At the start of each round, all miners start working on a new computation problem, derived from the last block, and the miner that is the first to solve the problem gets to write the next block. In Bitcoin, the difficulty of these computation problems is automatically adjusted by the protocol to get 1 new block roughly every 10 minutes. See [13] for further details on how blockchains work.

2.2 Namecoin's Naming System

Namecoin is one of the first forks of Bitcoin and is the oldest blockchain other than Bitcoin that is still operational, with a current market cap of 5 million USD [5] (the market cap of a cryptocurrency is the exchange-traded value of its coins multiplied by its number of coins in existence). The main motivation for starting Namecoin was to create an alternate DNS-like system that replaces DNS root servers with a blockchain for storing information on registered domain names [37]. Given that blockchains don't have central points of trust, a blockchain-based DNS is much harder to censor and registered names cannot be seized from owners without getting access to their respective private keys [29] Altering name registrations stored in a blockchain requires prohibitively high computing resources because re-writing blockchain data requires proof-of-work [7].

Namecoin is derived from Bitcoin's code, and keeps

most functionality identical to Bitcoin, with the exception of support for registering name/value pairs. **Our work on Blockstack implements name registration functionality as a separate layer *on top of* Bitcoin, but this design was non-obvious before our work.** Indeed, it was common practice to start alternate blockchains by forking them from Bitcoin and make modifications required by the respective service/application, the precise approach taken by Namecoin.

Just like DNS, there is a cost associated with registering a new name. The name registration fee discourages people from grabbing a lot of names that they don't intend to actually use. In Namecoin, the recipient of registration fees is a "black hole" cryptocurrency address from which money cannot be retrieved [29]. Namecoin defines a pricing function for how cost of name registrations change over time. Namecoin supports multiple namespaces (like TLDs in DNS), and the same rules for pricing and name expiration apply to all namespaces. By convention, the $d/$ namespace is used for domain names. For example, to register the domain *yahoo* on Namecoin, one must register the name $d/yahoo$ and then put the IP address of the Yahoo! website in the name/value pair.

In Namecoin, name registration is a two-step process. A user first **pre-orders** a *name* in a new transaction that includes $hash(name)$ in the transaction. This does not reveal what *name* she is trying to register. After the pre-order transaction has been confirmed by the network—i.e., enough blocks (typically 10) are later added to the blockchain to make it computationally infeasible for any miner to re-write recent blockchain history and reverse the transaction—the user can reveal the *name* she was actually trying to register. This is done by sending a second transaction on the network that completes the **register** step. The user includes the *value* of the name/value pair in the second transaction as well. The cryptocurrency address that signed the two transactions becomes the owner of the newly registered name/value pair.

Name registrations expire after a fixed amount of time, measured in new blocks written (currently 36,000 blocks, which translates to roughly 8 months). Namecoin also supports updating the value associated with a name, as well as ownership transfers.

2.3 Blockchain-based PKI system

We used Namecoin to build a PKI system, called *Blockchain IDs*, by starting a new namespace $u/$ on it. We defined the format for publishing public-keys, like PGP [49], along with other profile data in the blockchain [2]. This is similar to the format of DNS records. We launched a web service [39] in March 2014 that enabled people to easily register names on the $u/$ namespace of Namecoin and associate profile data with them. In our web service, we first register the name on the user's behalf (and also cover the cost of name registration for them) and then transfer the name to a cryptocurrency address owned by the user. Our implementation is the first PKI system that binds user identities to public-keys using a blockchain. All registered names have a ECDSA public-key [26] binding by default, and a subset of users added their PGP keys as well. More than 33,000 users registered on the Namecoin blockchain between March 2014 and August 2015 through our service. According to a study by Harry et al. [29], our system is the second largest namespace on Namecoin by volume and the largest by number of active users.

3 Lessons from Namecoin Deployment

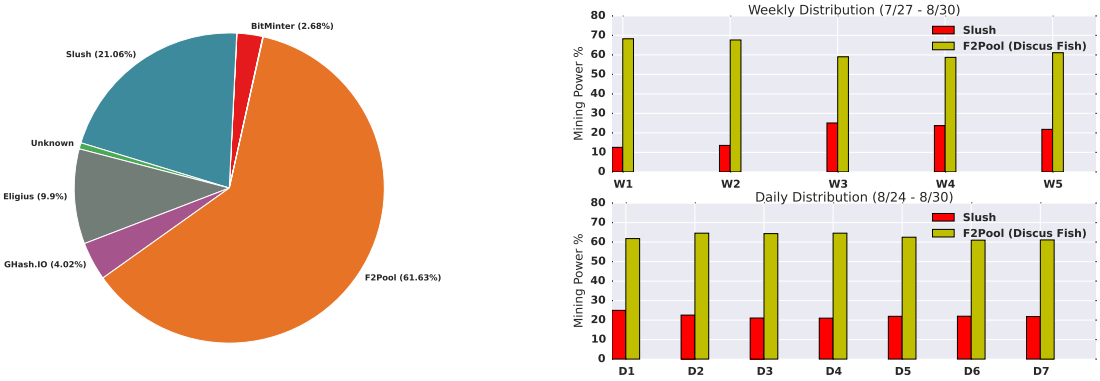
In this section, we describe our experience with running a year-long production system on Namecoin, the challenges we faced, and lessons we learned.

3.1 Security

The security of name ownership is tied to the security of both the underlying blockchain and the software powering it. There are three security issues to consider:

1) Cost of Attack: Miners often pool their resources to form a *mining pool*, which is essentially a super node on the network (a lot of computational power behind a single miner node). If the amount of computational power under the control of a single miner (or pool) is more than the rest of the network, called a 51% attack, then that miner has the ability to attack the network and rewrite recent blockchain history, censor transactions (e.g., for name registrations), and steal cryptocurrency using *double spend* attacks [45]. This is because it will win the leader election majority of the time, and produce a blockchain history with more proof-of-work than any disagreeing miner. The more expensive it is to control a majority of the compute power on a particular blockchain, the more secure the blockchain.

We noticed in late 2014 that a single mining pool consistently had more than 51% of the compute power on Namecoin. Recently, the situation has been even worse, with a single mining pool controlling over 60% of Namecoin's compute power. Figure 1 shows the monthly, weekly, and daily distribution of mining power for the month of August 2015, right before we migrated our system away from Namecoin. In fact, we have observed F2Pool (also known as Discus Fish) control up to 75% compute power in a particular week. At such concentration, Namecoin is effectively controlled by a single party; F2Pool gets to write most of the new blocks and can undermine the security of the blockchain at will.



(a) Monthly Mining Distribution

(b) Weekly and Daily Distribution

Figure 1: Mining Distribution (Aug 2015, the month before our migration away from Namecoin)

2) Software Vulnerabilities: Raw hashing power is not the only metric for the security of a blockchain. Software issues/bugs are also very important, e.g., a Namecoin bug allowed people to steal names from anyone [24]. Actively developed codebases with frequent security reviews lower the rate of critical bugs in production. Despite originally sharing the same codebase as Bitcoin, Namecoin was not kept up-to-date with advancements made in Bitcoin development.

3) DDoS Attacks: One of the less explored security issue with cryptocurrency blockchains is network attacks, e.g., DDoS attack on core discovery nodes, mining pools [34], or the entire network. The more peers a cryptocurrency network has, the more resilient the network is to denial-of-service attacks. As a relatively small network, Namecoin has many fewer nodes than Bitcoin (170 vs. 4,600 in Jan 2016 [3]), which makes it more vulnerable in this respect.

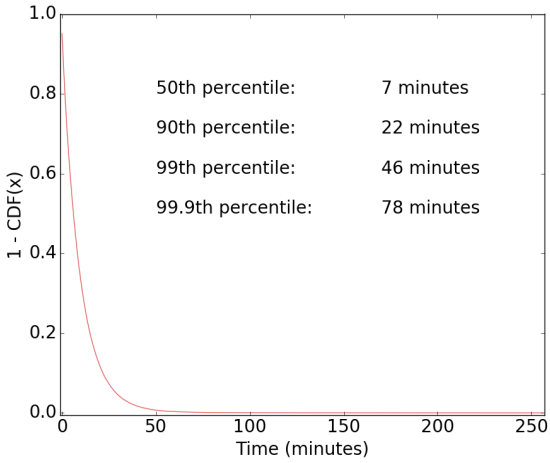
When considering security on these three metrics, the Bitcoin blockchain is currently by far the most secure blockchain. **We faced a fundamental tradeoff between security and introducing new functionality to blockchains.** Starting a new blockchain network is how developers introduce new functionality not provided by Bitcoin e.g., a naming system that is of interest to many emerging applications. However, new blockchains are significantly less secure than Bitcoin. In Section 4, we introduce Blockstack to show how to overcome this tradeoff by creating *virtual chains* to introduce new functionality as a layer on top of Bitcoin.

3.2 Network Reliability and Throughput

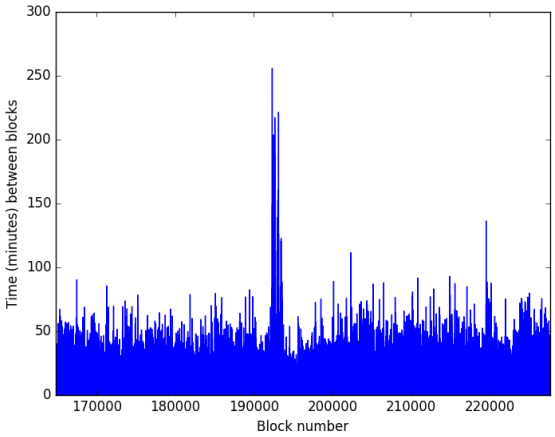
The throughput of our PKI system (number of entries we can register/update) is directly dependent on the throughput of the underlying blockchain. The number of new register/update operations that can be performed per hour is limited by the number of transactions that can be sent (and confirmed) on the underlying blockchain per hour. Similarly, reliability of our PKI system is impacted if the underlying blockchain cannot perform operations reliably and consistently.

Network Latency Spike: As a fork of Bitcoin, Namecoin shares many protocol properties with Bitcoin, including a 10 minute target of writing new blocks (latency target) and a 1MB bandwidth limit on block size (giving throughput of $\sim$1000 transactions per block). Figure 2(a) shows that since we launched our PKI system in March 2014, Namecoin on average performed well on the network latency target. As expected, most new blocks were written within 10 and 40 minutes (similar times have also been observed on Bitcoin [13]). Figure 2(b) shows an incident in late August 2014 (at block number 192000), where network latency skyrocketed for a couple of weeks ($\sim$1000 blocks are roughly a week). After investigating the issue and having discussions with Namecoin developers, we discovered that the latency spike was caused by software issues in Namecoin. Someone on the network was sending transactions with a large number of data fields per transaction. This was causing severe performance problems for the miners and their Namecoin daemons kept crashing. Without stable miner nodes there is no one there to package blocks, and hence the longer delay in new blocks. This shows that unexpected protocol/software issues can trigger network latency problems. During this period, we noticed a slow down in rate of new registrations of our PKI system along with a spike in user complaints and support requests.

Network Throughput Drop: In early September 2014, right after the latency spike incident we noticed that our transactions were not getting accepted for many



(a) CCDF of typical network latency (03/14 – 04/15)



(b) Network latency per new block (03/14 – 04/15)

Figure 2: Network Latency

consecutive blocks and, after a while, will get accepted in bulk in a single block that packaged a lot of transactions. We noticed that a lot of new blocks had no transactions in them. This issue persisted for over a week and Figure 3 plots the number of transactions that we were trying to send (shown as "tx target") vs. the number of transactions that were getting accepted by the network. Network latency was completely normal (shown at top of Figure 3), but network throughput went down because of no transactions in new blocks. We tried upgrading our software and rebroadcasting transactions, but the issue persisted. We concluded that there is a large mining pool that is either intentionally refusing or is unable to package transactions in the blocks it is writing. Our transactions will get packaged only when some other miner was elected to write the new block. We discuss this issue in more detail in the next section.

3.3 Potential Selfish Mining

The signs that we noticed in the incident where miners were not accepting our transactions (Section 3.2) looked similar to a selfish mining attack [20]. In a selfish mining attack, a) a miner needs to have a large amount of mining power (more than 33%), b) people would notice long delay in blocks followed by blocks in very quick succession, and c) there will be a lot of rejected blocks. We noticed all these signs, and believe that the unusually high computing power of a single miner was resulting in conditions similar to selfish mining i.e., the miner was able to work on new blocks faster than others and announce them in rapid succession. This is the first time that data collected from a production network shows signs of selfish-mining like behavior, regardless of if the miner was intentionally attacking the network or not.

3.4 Software Upgrades of Nodes

For updates to name pricing or other major changes, Namecoin requires a "hard fork" in which everyone on the network must upgrade their software, and nodes on previous versions can no longer participate in the blockchain network. Anecdotal evidence suggests that it's hard to get miners to upgrade their software because they don't have enough incentive to spend engineering hours on maintaining a small cryptocurrency like Namecoin, which is not their main reason for operating a mining pool. Our experience monitoring the Namecoin network showed that whenever software updates were issued on Namecoin, there was a considerable fluctuation of computing power. In fact, we noticed that after the recent upgrade to Namecoin Core [38], a major upgrade to the Namecoin daemon, many miners dropped out and never came back online.

We learned two operational lessons about software upgrades to cryptocurrency networks: a) **we should separate consensus breaking upgrades** from other upgrades as a software engineering rule (Bitcoin recently started doing this more cleanly in their codebase [10]) and b) **miners have incentives to minimize their cost (engineering time) of software upgrades**. This resistance to upgrades is present in Bitcoin as well, but is exaggerated for the "long tail" of smaller crypto-currencies. In Section 4 we describe how Blockstack accounts for these incentives and enables the ability to introduce new features without requiring miners to upgrade software.

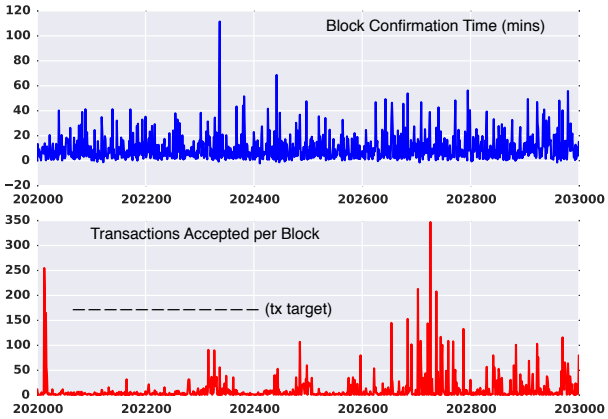


Figure 3: *Throughput, 'tx target' is the number of transactions we were trying to send*

3.5 Failure of Merged Mining

Security of a blockchain depends on relative compute power that miners have and how much would it cost a single party to get more computing power than the rest of the network. New, smaller blockchains have a *bootstrapping problem* where in the initial days of a new blockchain it would be relatively easy for a single party to take it over, since the total compute power on the blockchain is not yet large enough to prevent this. To address this problem Satoshi Nakamoto, author of Bitcoin, introduced "merged mining" [35] where an alternate blockchain can allow Bitcoin miners to participate in the new network without requiring them to spend extra compute cycles. The miners can make extra profits on the new blockchain without adding computational overhead. With a merge-mined cryptocurrency the security of the blockchain is typically a subset of the "main blockchain", because in practice not all miners of the main blockchain go through the trouble of setting up merged mining.

Namecoin switched to merged mining with Bitcoin to increase security of the blockchain [29]. Namecoin is the oldest and largest merged-mined cryptocurrency and inspired other cryptocurrencies to consider it as well. One of our key findings is that merged mining is currently failing in practice: the leading merged-mined blockchain, Namecoin, is vulnerable to the 51% attack (Section 3.1). Moreover, merged-mining provided a false sense of security to the Namecoin community. If Namecoin shares compute power of Bitcoin, that might be a high compute power in absolute terms but a single miner can still have more than 51% compute power on the merged-mined blockchain. The security of the blockchain depends on *relative* compute power and not on absolute compute power. F2pool, for example, had 30-35% compute power of Bitcoin, when it had more than 60% compute power of Namecoin. Unless the merged mined cryptocurrency can consistently get a very high ratio of main blockchain miners to support their software, merged mining will not keep it safe from 51% attacks.

The merged mining failure of Namecoin convinced us that **we're at a stage in the evolution of blockchains where there is not yet enough compute cycles dedicated to mining to support multiple secure blockchains**. The respective financial capital attached to blockchains relative to Bitcoin supports this argument: Bitcoin has a 5.9 billion USD market cap, which accounts for 89% of the market cap of all 500+ blockchains combined, while the 2nd and 3rd largest market caps are 3.2% and 2.6% of Bitcoin respectively [5]. It's possible that after some years or decades we might have multiple secure blockchains, but in the near-future Bitcoin's blockchain is the only one that is prohibitively expensive to attack.

3.6 Summary

Namecoin deserves full credit for originally solving naming on a blockchain. But after considering all of the above factors, it was an easy decision to move our PKI system from Namecoin to Bitcoin. In general, after our experience with running the production network, we strongly believe that decentralized applications and services need to be on the largest, most secure, and most actively maintained blockchain. Currently, no other blockchain even comes close to Bitcoin in terms of these security requirements.

4 Design of Blockstack

Blockstack is designed to implement a naming system with human-readable names in a layer above the blockchain. In this section, we describe how Blockstack uses the underlying blockchain, and present how it copes with technical limitations of contemporary blockchains.

4.1 Challenges

Building systems with blockchains presents challenges:

• **Limits on Data Storage:** Individual blockchain records are typically on the order of kilobytes [45] and cannot hold much data. Moreover, the blockchain's log structure implies that *all* state changes are recorded in the blockchain. All nodes participating in the network need to maintain a full copy of the blockchain, limiting the total size of blockchains to what current commodity hardware can support. As of January 2016, Bitcoin nodes

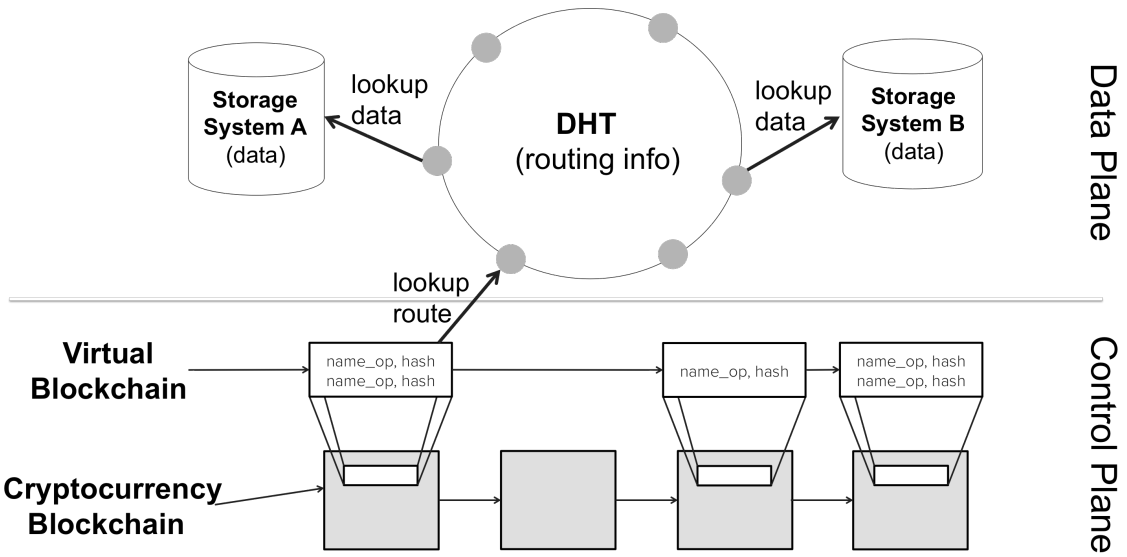


Figure 4: *Overview of Blockstack's architecture. Blockchain records give (name, hash) mappings. Hashes are looked up in a DHT to discover routes to data. Data, signed by name owner's public-key, is stored in cloud storage.*

need to dedicate 53GB total disk space to blockchain data for staying synchronized with the network.

• **Slow Writes:** The transaction processing rate is capped by the blockchain's write propagation and leader-election protocol, and it is pegged to the rate at which new blocks are announced by leader nodes (called *miners* in blockchain networks [13]). New transactions can take several minutes to a few hours to be accepted.

• **Limited Bandwidth:** The total number of transactions per block is limited by the *block size* of blockchains. To maintain fairness and to give all nodes a chance to become leader in the next round, it's required that all nodes receive a newly announced block at roughly the same time. Therefore, the *block size* is typically limited by average uplink bandwidth of nodes in the network [13]. For Bitcoin the current bandwidth is 1MB ($\sim$1000 transactions) per new block.

• **Endless Ledger:** Integrity of blockchains depends on the ability for anyone to audit them back to the first block. As the system makes forward progress and issues new blocks, the cost of this auditing grows linearly with time and booting up new nodes becomes progressively more time consuming. We call this the *endless ledger problem*. Bitcoin's blockchain currently has $\sim$395,000 blocks and new nodes take 1-3 days to download blockchain from Bitcoin peers, verify it, and bootup.

4.2 Architecture Overview

Blockstack builds a naming system as a separate logical layer on top of the underlying blockchain. Blockstack uses the underlying blockchain to achieve consensus on the state of this naming system. It uses the underlying blockchain as a communication channel for announcing state changes; any changes to the state of the naming system can only be announced in new blockchain blocks. Relying on the consensus protocol of the underlying blockchain, Blockstack can give total ordering for all operations (like register, update, and transfer) supported by the naming system.

Separation of Control and Data Plane: Blockstack decouples security of name registration and name ownership from data availability of values associated with names. This is done by separating the control and data planes (Figure 4). The control plane is responsible for registering human-readable *names* and creating $(name, hash)$ bindings. It also defines the protocol for establishing ownership of names, which are owned by cryptographic keypairs. The control plane consists of a cryptocurrency blockchain and a logically separate layer on top, called a "virtual blockchain" (Section 4.3.2).

The data plane is responsible for data storage and availability. It consists of (a) routes for discovering data, and (b) external storage systems for storing data (such as Amazon S3, IPFS [27], or Syndicate [28]). Data values are signed by public keys of respective owners. The control plane retrieves data values from the data plane and verifies their authenticity by checking either the hash of the data or the signature of the public key.

We believe this separation is a significant improvement over Namecoin, which implements both control and data plane at the blockchain level. Our design not

only significantly increases the data storage capacity of the naming system, but also allows each layer to evolve and innovate independently of each other.

Agnostic of Underlying Blockchain: The design of Blockstack does not put any limitations on which cryptocurrency blockchain can be used with it. Any blockchain can be used, but the security and reliability properties are directly dependent on the underlying blockchain. We believe that the ability to *migrate* from one blockchain to another is an important design choice as it allows for the larger system to survive, even when the underlying blockchain is compromised. Currently, Blockstack core developers decide which underlying blockchain(s) to support in which version of the software. Individual applications can decide to run the software version of their choice and keep their namespace on a particular blockchain, if they prefer to not migrate. Section 5 gives more details on the migration process.

Ability to Construct State Machines: A key contribution of Blockstack is the introduction of a logically separate layer on top of a cryptocurrency blockchain that can construct an arbitrary *state machine* after processing information from the underlying blockchain. We call this layer a *virtual blockchain* (Section 4.3.2). A virtual blockchain treats transactions from the underlying blockchain as inputs to the state machine. Valid inputs trigger state changes. At any given time, where time is defined by the block number, the state machine can give exactly one global state. Time moves forward as new blocks are written in the underlying blockchain and the global state is updated accordingly.

Virtual blockchain can introduce new types of state machines without requiring any changes from the underlying blockchain. Introducing new state machines directly in a blockchain requires, potentially consensus breaking, upgrades of the cryptocurrency blockchain that can cause forks and are hard to do in practice [13]. Currently, Blockstack introduces a state machine that represents the global state of a naming system i.e., who owns a particular name, what data is associated with a name etc. It's possible to use the *virtual blockchain* concept to define other types of state machines as well.

4.3 Blockstack Layers

Blockstack introduces new functionality on top of blockchains by defining a set of new operations that are otherwise not supported by the blockchain. Blockstack has four layers, with two layers (cryptocurrency blockchain and virtual blockchain) in the control plane and two layers (discovery and storage) in the data plane.

4.3.1 Layer 1: Cryptocurrency Blockchain

The blockchain occupies the lowest tier, and serves two purposes: it stores the sequence of Blockstack operations, and provides consensus on the order in which they were written. Blockstack operations are encoded in transactions on the underlying blockchain.

4.3.2 Layer 2: Virtual Blockchain

Above the blockchain is a *virtual blockchain*, also called *virtualchain*. The main advantage of virtualchains is to introduce new functionality/operations without requiring changes to the underlying blockchain. Only Blockstack nodes are aware of this layer and underlying blockchain nodes are agnostic to it. Blockstack operations are defined in the virtualchain layer and are encoded in valid blockchain transactions as additional metadata. Blockchain nodes do see the raw transactions, but the logic to process Blockstack operations only exists at the virtualchain level.

The rules for accepting or rejecting Blockstack operations are also defined in the virtualchain. Accepted operations are processed by the virtualchain to construct a database that stores information on global state of the system along with state changes at any given blockchain block. Virtualchains can be used to build a variety of state machines. Currently, Blockstack defines only a single state machine; a global naming system.

4.3.3 Layer 3: Discovery Layer

Blockstack separates the task of *routing* requests (i.e., how to discover data) from the actual storage of data. This avoids the need for the system to adopt a single or particular storage service from the onset, and instead allows multiple storage providers to coexist and compete for use (including both commercial services and peer-peer systems).

The virtualchain binds *names* with respective $hash(route)$ and stores these bindings in the control plane, where as the actual routes are stored in the discovery layer. **Users do not need to trust the discovery layer**, because the integrity of routes can be verified by checking the $hash(route)$ in the control plane.

In Blockstack's current implementation, nodes form a DHT-based peer network [33] for storing routes. The DHT only stores routes if $hash(route)$ was previously announced in the blockchain and effectively white-lists the data that can be stored in the DHT. Due to space constraints, we omit most details of our DHT storage from this paper; the key aspect relevant to the design of Blockstack is that routes (no matter from where they are fetched) can be verified and therefore cannot be tampered with.

4.3.4 Layer 4: Storage Layer

The top-most layer is the storage layer, which hosts the actual data values of name/value pairs. All stored data values are signed by the key of the respective owner of a *name*. By storing data values outside of the blockchain, Blockstack allows values of arbitrary size, and allows for a variety of storage backends. **Users do not need to trust the storage layer**, because they can verify the integrity of the data values in the control plane. There are two modes of using the storage layer and they differ in how the integrity of data values is verified; Blockstack supports both storage modes simultaneously.

a) Mutable Storage is the default mode of operation for the storage layer. Bindings between *name* and $hash(route)$ are kept in the control plane and the discovery layer is used to discover data values. Data integrity is verified by validating the signatures associated with values. Mutable storage allows for faster writes, since data updates do not involve any transactions on the blockchain, which are slow. Updates to name/value pairs do not involve the blockchain.

b) Immutable Storage by-passes the discovery layer and stores bindings between *name* and $hash(data)$ in the control plane, instead of bindings between *name* and $hash(route)$. Data integrity is verified by hashing the data and comparing the result against the $hash(data)$ from the control plane. This mode is suitable for data values that don't change often and where it is important to verify that you are viewing the latest version of the data value. For immutable storage, updates to data values require a new transaction on the underlying blockchain, making data updates much slower than mutable storage.

4.4 Naming System

Blockstack uses the four-tiered system to implement a complete naming system. Names are owned by cryptocurrency addresses of the underlying blockchain and the associated private keys e.g., ECDSA-based private keys used in Bitcoin [13]. As with Namecoin, a user *preorders* and then *registers* a name in two steps, in order to claim a name without revealing it to the world first (otherwise an attacker can race the user in claiming the name). The first user to successfully write both a preorder and a register transaction owns the name. Any previous preorders become invalid when a name is registered. Once registered, a user can *update* the name/value pair by sending a update transaction (which changes the name/value binding) and uploading the new value to the storage layer. Name *transfer* simply changes the cryptocurrency address that is allowed to sign subsequent transactions *revoke* disables any further operations.

The naming system is implemented by defining a state machine and rules for state transitions in the virtualchain. Figure 5 shows the different states a *name* can be in and how state transitions work. Names are organized into *namespaces*. A namespace is the functional equivalent of a top-level domain in DNS—it defines the cost of names, and the name's renewal rate. Like names, a namespace must be preordered and then registered.

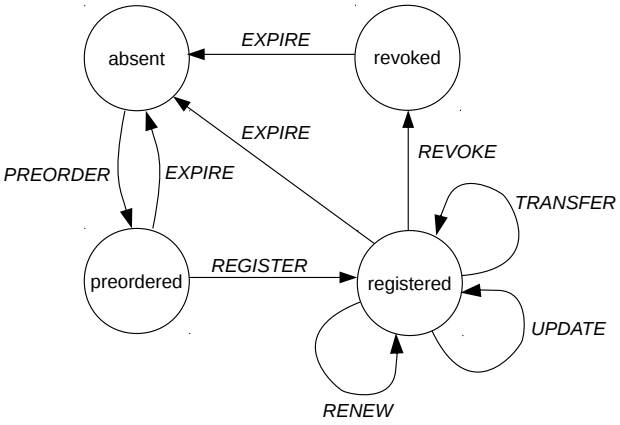


Figure 5: *States and Transitions for a Name.*

4.4.1 Pricing Functions for Namespaces

Anyone can create a namespace or register names in a namespace, as there is no central party to stop someone from doing so. *Pricing functions* define how expensive it is to create a namespace or to register names in a namespace. Defining intelligent pricing functions is a way to prevent "land grabs" and stop people from registering a lot of namespaces/names that they don't intend to actually use. Blockstack enables people to create namespaces with sophisticated pricing functions. For example, we use the *.id* namespace for our PKI system and created the *.id* namespace with a pricing function where (a) the price of a name drops with an increase in name length and (b) introducing non-alphabet characters in names also drops the price. With this pricing function, price of *john.id* $>$ *johnadam.id* $>$ *john0001.id*. The function is generally inspired by the observation that short names with all-alphabets are considered more desirable on namespaces like Twitter. It's possible to create namespaces where name registrations are free as well. Further, we expect that in the future there can be a reseller market for names, just like DNS. A detailed discussion of pricing functions is out of the scope of this paper and the reader is encouraged to see [29] for more details on pricing functions.

4.5 Simple Name Verification

Blockstack nodes can independently calculate a *consensus hash* at any blockchain block. Consensus hashes help Blockstack nodes figure out if they have the same view of the global state at any given block. Each consensus hash $CH(h)$ is constructed from block h's sequence of virtualchain operations V_h, as well a geometric series of prior consensus hashes P_h defined by:

$$CH(h) = hash(V_h + P_h)$$

where,

$$P_h = \{CH(h-2^i) | i \in \mathbb{N}, h-2^i >= h_0\}$$

and h_0 is the first block. Other than detecting that two Blockstack nodes have the same global view, consensus hashes also address the *endless ledger problem* (defined in Section 4.1). As the underlying blockchain grows in size, new Blockstack nodes need to process more and more blocks before they bootup.

A new Blockstack node can bootstrap by using an untrusted database of state information at a given block number, combined with a trusted consensus hash $CH(h)$ of the same block number. The block number is also termed the *block height* in the literature, and it increases with each new block. A new Blockstack node can reconstruct the virtual blockchain from the untrusted database and reprocess virtualchain operations at each blockchain block. The new node can also calculate $CH(h)$ at each blockchain block. If the final consensus hash matches the trusted consensus hash at h_n, then the database associated with h_n is trustworthy and the node can start processing blocks after h_n. This is much faster than the traditional approach of starting from the first block h_0 and fetching all transaction even though most will be discarded.

The process of verifying the authenticity of a prior name operation with a later trusted consensus hash is called *Simplified Name Verification (SNV)*. SNV enables support for "thin clients", who can query the past state of the system without running Blockstack nodes or having access to the full blockchain history. Support for thin clients is important for users on mobile devices.

As such, if a user trusts that $CH(h)$ is authentic, then she can query and verify the virtualchain operations V_h and previous consensus hash P_h for block h. The construction of $CH(h)$ allows a user to verify the authenticity of any virtualchain operation from a block with height $h_{prior} < h$, using only a logarithmic number of queries. Figure 6 shows an example SNV query. Each row represents the blockchain, in decreasing block height order from left to right ($h > h_0$). The user wishes to verify the authenticity of a name operation in a target block (marked with a ?). In each step, the user recursively trusts the consensus hash for the white outlined blocks.

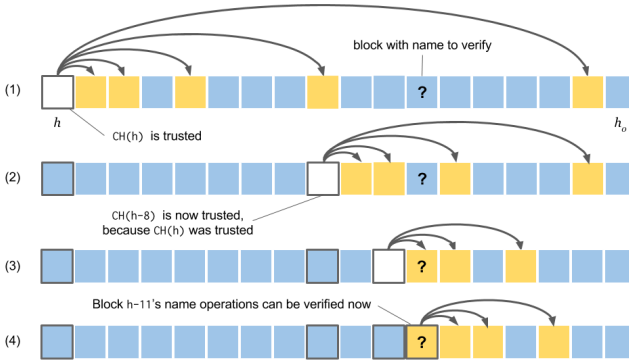


Figure 6: *Overview of SNV. Example SNV query.*

On current commodity hardware, booting new Blockstack nodes can take 1-2 hours with SNV, compared to 2-4 days without SNV. Further engineering improvements in our Python implementation are currently possible.

5 Lessons from Migration to Bitcoin

We implemented Blockstack in 40,344 lines of Python code [12] and the current implementation uses Bitcoin as the underlying blockchain. In September 2015, we completed migration of 33,000 users of our production PKI system [39], from Namecoin to Blockstack/Bitcoin. These users were migrated from the *u/* namepsace on Namecoin to the *.id* namespace on Blockstack.

Blockstack embeds additional data in Bitcoin transactions using special fields dedicated for including arbitrary data [11]. Embedding additional data in Bitcoin transactions is already becoming a popular way to define higher-level protocols on top of Bitcoin, like Counterparty [18], Open Assets [40], etc. Figure 7 shows recent bandwidth usage of data-embedding protocols, like Blockstack, on the Bitcoin blockchain. The spike of 10,000+ transactions, near block 375000, was during our migration to Bitcoin. Our production system [39] currently accounts for most of Blockstack transactions, which is 24.4% of all data-embedding transactions ever made on Bitcoin [41]. Below are some observations we made while working with the Bitcoin network:

• **Network Throughput:** The Bitcoin developers and community is currently going through a (heated) debate about increasing the blocksize limit from 1MB to 8MB [48]. Bitcoin currently supports roughly 7 transactions per second with a 1MB block size. We noticed these limitations first hand when we throttled our transactions to not exceed 20-30% of Bitcoin blocks, significantly increasing the amount of time it took for completing registrations. When scaling to millions of users, instead of thousands, even 8MB blocks will not suffice

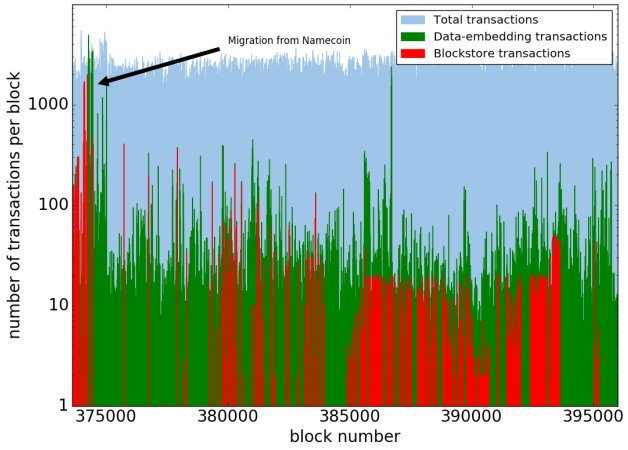


Figure 7: *Data-embedding transactions on Bitcoin are already becoming a frequent use case.*

and the community needs to look into side chains [9] and novel methods for packing multiple transactions in one (area of future work for us).

• **Network Attacks:** During our migration to Bitcoin, a UK based company CoinWallet was performing a stress test on the Bitcoin network [15]. The stress test included a high-volume of small transactions which were too small for miners to package in a block. This resulted in extremely high number of unconfirmed transactions on the network and we ended up paying 2-3 times higher transaction fees to get our transactions packaged by miners. This experience shows how a single actor can force high mining fees on the rest of the network (although in this case there was a cost factor attached to the attack). We believe that networking attacks, like the one we experienced or other DDoS attacks [34], are likely to become more frequent as the Bitcoin network is used for more mainstream services. Protections against such attacks is an important area of future research.

6 Related Work

Binding names to values in naming systems is a well-explored problem space. UIA [22] gives a great overview of global naming systems and their importance. Unlike Namecoin [37] or Blockstack, UIA doesn't try to provide globally unique names. We encourage the reader to UIA [22] for a detailed background on naming systems. In authentication systems like InCommon [25], OpenID [43], and the Web's certificate authorities, a federation of authorities attests to bindings. Blockstack does not require a federation.

Other than Namecoin, blockchains like Ethereum [6] and BitShares [4] also have support for namespaces. Further, sidechains [9] enable implementation of naming systems as an alternate blockchain that is linked to the main Bitcoin blockchain. All these designs involve smaller, alternate blockchains and Blockstack implements a naming system directly on top of the Bitcoin.

In networked systems it's hard to get global state without involving central/trusted parties [31], Blockstack is able to give global state (and not just approximate global state). Our system is open ("permissionless"), whereas existing wide-area systems like OceanStore [19] and Bonafide [14] have a closed ("permissioned") set of peers that use BFT agreement to make progress for the whole system. Blockstack differs from decentralized storage systems which allow open membership but offer stronger-than-eventual data consistency (like Shark [8], Pond [44], and Scatter [23]) by focus on decentralization while supporting a wide variety of external datastores that give strong consistency.

Storage-oriented cryptocurrencies like Filecoin [21], Permacoin [36], and Storj [46] seek to replace cloud storage by distributing files as sets of transactions within a blockchain, and rewarding miners for proof-of-storage (instead of proof-of-work). Blockstack differs from these systems by decoupling hosting data from operations of the underlying cryptocurrency blockchain, allowing developers to use storage systems appropriate for their problem domains. Blockstack currently uses a simple Kademlia [33] based DHT as discovery layer, but other protocols like Chord [47] or caching optimizations like Beehive [42] are possible.

7 Conclusion

Our experience with running a production network on Namecoin, one of the oldest and largest cryptocurrency blockchains other than Bitcoin, shows how a single miner consistently had more than 51% hashing power and how network reliability was far inferior to Bitcoin. Our data shows that out of the hundreds of blockchains currently in use [5] even the more stable and more popular blockchains like Namecoin are not suitable for production use. Currently, the security of Bitcoin far out weights other blockchains.

We have presented Blockstack, a blockchain-based naming system that separates control and data planes and enables the ability to introduce new functionality without modifying the underlying blockchain. The design of Blockstack was informed by a year of production experience from one of the largest blockchain-based systems. We've made several improvements (faster bootstrapping of new nodes, keeping data updates off the slow blockchain network, etc.) that make it easier to build decentralized services using publicly-available infrastructure. We've released Blockstack as open-source [12].

References

[1] Bitcoin hashrate. https://blockchain.info/charts/hash-rate.

[2] Blockchain id format, ver 2. https://github.com/blockstack/blockchain-id/wiki/Profile-Schema-v2.

[3] Crypto-currencies statistics – active nodes. https://bitinfocharts.com.

[4] Bitshares namespaces, 2016. http://docs.bitshares.eu/namespaces/index.html.

[5] Cryptocurrency market cap, Jan. 2016. http://www.coincap.io.

[6] A next-generation smart contract and decentralized application platform, 2016. https://github.com/ethereum/wiki/wiki/White-Paper.

[7] Adam Back. Hashcash - A Denial of Service Counter-Measure. Tech report, 2002. http://www.hashcash.org/papers/hashcash.pdf.

[8] S. Annapureddy, M. J. Freedman, and D. Mazieres. Shark: Scaling file servers via cooperative caching. In *Proc. 2nd NSDI*, Boston, MA, 2005.

[9] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timon, and P. Wuille. Enabling Blockchain Innovations with Pegged Sidechains. White paper, Blockstream, 2014. https://blockstream.com/sidechains.pdf.

[10] Bitcoin Core Developers. Bitcoin core version 0.10.0 release notes: Bitcoin consensus library, Feb. 2015. https://bitcoin.org/en/release/v0.10.0.

[11] Bitcoin.org: Bitcoin developer guide, 2015. http://bitcoin.org/en/developer-guide.

[12] Blockstack source code release v0.10, 2016. http://github.com/blockstack/blockstack-server.

[13] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 104–121, 2015.

[14] B.-G. Chun, P. Maniatis, S. Shenker, and J. Kubiatowicz. Tiered fault tolerance for long-term integrity. In *FAST*, pages 267–282, 2009.

[15] Coindesk. Bitcoin network stress test could occur next week, Sep 2015. http://coinde.sk/1Ku5oWc.

[16] Coindesk. State of bitcoin 2015: Ecosystem grows despite price decline, 2015. http://coinde.sk/1tJDDvv.

[17] Coindesk. Bitcoin venture capital, Jan. 2016. http://www.coindesk.com/bitcoin-venture-capital/.

[18] Counterparty protocol specifications. http://counterparty.io/docs/protocol_specification/.

[19] P. Eaton, H. Weatherspoon, and J. Kubiatowicz. Efficiently binding data to owners in distributed content-addressable storage systems. In *Security in Storage Workshop, 2005. SISW'05. Third IEEE International*, pages 12–pp. IEEE, 2005.

[20] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *CoRR*, abs/1311.0243, 2013.

[21] Filecoin: A Cryptocurrency Operated File Network. Tech report, 2014. http://filecoin.io/filecoin.pdf.

[22] B. Ford, J. Strauss, C. Lesniewski-Laas, S. Rhea, F. Kaashoek, and R. Morris. Persistent personal names for globally connected mobile devices. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI '06)*, Seattle, Washington, Nov. 2006.

[23] L. Glendenning, I. Beschastnikh, A. Krishnamurthy, and T. Anderson. Scalable consistency in scatter. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 15–28. ACM, 2011.

[24] M. Gronager. Namecoin was stillborn, i had to switch off life-support, Oct 2013. https://bitcointalk.org/index.php?topic=310954.

[25] Incommon federation. https://www.incommon.org/federation/.

[26] D. Johnson, A. Menezes, and S. A. Vanstone. The elliptic curve digital signature algorithm (ecdsa). *Int. J. Inf. Sec.*, 1(1):36–63, 2001.

[27] Juan Benet. IPFS - Content Addressed, Versioned, P2P File System. Draft, ipfs.io, 2015. https://github.com/ipfs/papers.

[28] Jude Nelson and Larry Peterson. Syndicate: Virtual cloud storage through provider composition. In *ACM BigSystem*, 2014.

[29] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, and A. Narayanan. An empirical study of Namecoin and lessons for decentralized namespace design. *WEIS '15: Proceedings of the 14th Workshop on the Economics of Information Security*, June 2015.

[30] D. Kaminsky. Spelunking the triangle: Exploring aaron swartzs take on zookos triangle, Jan 2011. http://dankaminsky.com/2011/01/13/spelunk-tri/.

[31] S. Keshav. Efficient and decentralized computation of approximate global state. *SIGCOMM Comput. Commun. Rev.*, 36(1):69–74, Jan. 2006.

[32] J. A. Kroll, I. C. Davey, and E. W. Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *WEIS 2013*.

[33] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 53–65, London, UK, UK, 2002. Springer-Verlag.

[34] J. McGovern. Official statement on the last weeks ddos-attack against ghash.io mining pool, 2015. http://bit.ly/1nu49vR.

[35] Merge mining specification. https://en.bitcoin.it/wiki/Merged_mining_specification.

[36] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz. Permacoin: Repurposing bitcoin work for data preservation. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 475–490. IEEE, 2014.

[37] Namecoin. https://namecoin.info.

[38] Namecoin core, required software for miners, 2015. https://github.com/namecoin/namecoin-core.

[39] Onename. https://onename.com.

[40] Open assets protocol. http://www.openassets.org.

[41] Statistics of usage for bitcoin op_return. http://opreturn.org.

[42] V. Ramasubramanian and E. G. Sirer. Beehive: O(1)lookup performance for power-law query distributions in peer-to-peer overlays. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, pages 8–8, Berkeley, CA, USA, 2004. USENIX Association.

[43] D. Recordon and D. Reed. Openid 2.0: A platform for user-centric identity management. In *Proceedings of the Second ACM Workshop on Digital Identity Management*, DIM '06, pages 11–16, New York, NY, USA, 2006. ACM.

[44] S. C. Rhea, P. R. Eaton, D. Geels, H. Weatherspoon, B. Y. Zhao, and J. Kubiatowicz. Pond: The oceanstore prototype. In *FAST*, volume 3, pages 1–14, 2003.

[45] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Tech report, 2009. https://bitcoin.org/bitcoin.pdf.

[46] Shawn Wilkinson and Tome Boshevski and Josh Brandof and Vitalik Buterin. Storj: A peer-to-peer cloud storage network. Tech report, storj.io, 2014. http://storj.io/storj.pdf.

[47] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *IEEE Transactions on Networking*, 11, Feb. 2003.

[48] A. V. Wirdum. A closer look at bip100: The block size proposal bitcoin miners are rallying behind, Aug. 2015. http://bit.ly/1VbyoXP.

[49] P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, USA, 1995.